Problem Set 4

Due Friday, May 2 at 1:00 PM Pacific

> *Solutions are available!*
>
> 🔑 **Solutions**

This fourth problem set explores graph theory and the pigeonhole principle. It's designed to take you on a tour of the beauty of finite structures and some of their amazing properties. Plus, you'll get to see some pretty pictures and learn about why all this matters in the first place. 😃

Good luck, and have fun!

Starter Files

Here are the starter files for the problems you'll submit electronically:

> 📦 PS4 Starter Files

Unpack the files somewhere convenient and open up the bundled project.

If you would like to type your solutions in $\LaTeX$, you may want to download this template file where you can place your answers:

> 🖊 PS4 $\LaTeX$ Template

General Advice

- First, read through the index of problems below to get an idea of time management for this assignment.
- Next, glance through all the questions before you start working. This not only ensures you won't be caught off guard as you reach later problems, but also gives your brain a chance to start working on some of these as a background process while you do other things.
- If you get stuck, consider moving on to another problem for a while! Changing things up is a great way to get out of a rut, make new connections between concepts, and spark creative insights!
- If you realize you don't have the requisite knowledge to complete a problem, take a step back from it and review the relevant lecture materials. Skipping lecture and trying to extract the relevant tidbits for each problem will lead to a more frustrating experience than engaging fully with a lecture and approaching the problem sets afterward.

Timeline

This pset has **three** required problems (one **autograded** and two **written / manually graded**).

This pset is scoped to take less time than most because we know you have a lot on your plate as you prepare for Midterm 1. Because there are so few questions, we are listing them all on a single page.

Due Friday, May 9 at 2:00 PM Pacific We recommend the following timeline for completing questions:

- Q1 and Q2 by Wednesday evening
- Q3 by Thursday evening

Of course, we recommend reading through the entire problem set ASAP so you have a strong idea of how much time the problems will take and so your brain can start working on them as a background process. :)

Problem One: Drawing Graphs

The starter files for this assignment contain a graph editor. Once you've loaded a graph file, you can add nodes by double-clicking in blank space. You can then add edges by hovering over a node and clicking and dragging from the node's blue border to the destination node.

In this problem, we're going to ask you to use the graph editor to draw a series of graphs. Some are specific, concrete graphs where the goal will be to interpret the notation you're presented with. Others will give a series of requirements on the graph and then ask you to work out what graph has those properties. By the time you've finished, we hope you'll have a better intuition for how graphs work and how they're structured.

    i. As a refresher, if $n \in \mathbb{N}$, then $[n] = \{k \mid k \in \mathbb{N} \wedge k < n\}$.

Consider the graph G defined below:

$$G = ([6], \{\{u, v\} \mid u, v \in [6] \wedge u \not\equiv_3 v\}).$$

Use the graph editor to draw this graph in the file **res/Part1.graph**.

Remember that a graph is defined as a pair where the first entry is the set of nodes and the second entry is the set of edges.

    ii. There is a connected graph where every node has degree 2, except for one node which has degree 4. Additionally, every cycle in the graph has length 4. This information uniquely defines the shape of the graph. Use the graph editor to draw this graph in the file **res/Part2.graph**.

    iii. There is a connected graph with five nodes that has an independent set of size four. This information uniquely defines the shape of the graph. Use the graph editor to draw this graph in the file **res/Part3.graph**.

    iv. There is a graph with five nodes. For each $n \in \{1, 2, 3, 4\}$, there is at least one node whose degree is exactly n. This information uniquely defines the shape of the graph. Use the graph editor to draw this graph in the file **res/Part4.graph**.

Problem Two: Properties of Directed Graphs

Let $G = (V, E)$ be a *directed* graph. Here are three possible properties that G could have:

    (I) $\forall x \in V. (x, x) \notin E.$
    (II) $\forall x \in V. \forall y \in V. ((x, y) \in E \to (y, x) \notin E).$
    (III) $\forall x \in V. \forall y \in V. \forall z \in V. ((x, y) \in E \wedge (y, z) \in E \to (x, z) \in E).$

Prove that if G has properties (I) and (III), then G also has property (II).

Problem Three: Bipartite Graphs

The ***bipartite graphs*** are an important and common family of graphs. They appear in a huge number of settings – error-correcting codes, computational social choice theory, and algorithm design, to name a few.

Let's start with a formal definition. An undirected graph $G = (V, E)$ is called **bipartite** if there exist two sets V_1 and V_2 such that the following are true:

(I) $\forall v \in V. \, (v \in V_1 \lor v \in V_2)$.

(II) $\forall v \in V. \, (v \notin V_1 \lor v \notin V_2)$.

(III) $\forall u \in V. \, \forall v \in V. \, (\{u, v\} \in E \to (u \in V_1 \leftrightarrow v \in V_2))$.

The sets V_1 and V_2 here are called **bipartite classes** of G.

    i. Suppose you have an 8×8 chessboard. We form a graph from the board as follows: there's a node for each square on the board, and an edge between any pair of squares that are immediately adjacent in one of the four cardinal directions (up, down, left, and right).

    Explain why this is a bipartite graph by telling us what the bipartite classes V_1 and V_2 are and briefly explaining why properties (I), (II), and (III) hold for those choices of V_1 and V_2. We're expecting at most a sentence of explanation for each of the three parts, and no formal proof is required.

    Don't do this in your head – draw lots of pictures!

Bipartite graphs have many interesting properties. One of the most fundamental is this one:

> **Theorem:** An undirected graph G is bipartite if and only if it contains no closed walks of odd length.

The forward direction of this implication has a nice intuition.

    ii. Explain, intuitively, why if G is bipartite, then it has no closed walks of odd length. Specifically, give us a brief, informal explanation as to why every closed walk in G has to have even length.

    It might help to draw some pictures.

The reverse direction of this implication – that if G has no closed walks of odd length, then G is bipartite – requires a different sort of argument.

Let's pick some arbitrary graph $G = (V, E)$ that has no closed walks of odd length. For simplicity's sake, we'll assume that G has just one connected component. If G has two or more connected components, we can essentially treat each one of them as independent graphs. *(Do you see why?)*

Now, choose any node $x \in V$. Using node x as an "anchor point," we can define two sets V_1 and V_2 that we'll need for the remainder of this argument:

$$V_1 = \{v \in V \mid \text{there is an odd-length walk from } x \text{ to } v\}$$

$$V_2 = \{v \in V \mid \text{there is an even-length walk from } x \text{ to } v\}$$

This turns out to be a really useful way to group the nodes of G.

    iii. Prove that V_1 and V_2 satisfy requirement (I) of bipartite graphs.

    Remember that there might be multiple different walks of different lengths from v to some other node x, so be careful not to talk about "the" walk between v and x. Also note that these walks are not necessarily paths.

    iv. Prove that V_1 and V_2 satisfy requirement (II) of bipartite graphs.

    v. Prove that V_1 and V_2 satisfy requirement (III) of bipartite graphs.

Your results from parts (iii), (iv), and (v) of this problem establish that G must be bipartite.

Want to learn more about bipartite graphs and their applications? Take CS161 (algorithms), CS224W (deep learning on graphs), or CS250 (error-correcting codes)!

Due Friday, May 2 at 1:00 PM Pacific

Optional Fun Problem: Forced Triangles

Let $G = (V, E)$ be a graph where each node has been given one of three different colors (ecru, puce, and zaffre — and yes, those are all colors) such that no two nodes of the same color are adjacent to one another. Furthermore, suppose there are exactly n nodes of each color.

Prove that if every node $v \in V$ has degree at least $n + 1$, then G contains a triangle (a copy of K_3).

All course materials © Stanford University 2025.

Website programming by Julie Zelenski with minor edits by Keith Schwarz and Sean Szumlanski. This page last updated 2025-May-05.